

Design and Implement a Reliable Release Process



Daniel Krzyczkowski

MICROSOFT MVP & SOFTWARE DEVELOPER

@DKrzyczkowski www.techmindfactory.com



Module Overview



Design and implement release gates and approval processes

Setup deployment to different cloud environments

Organize shared release configuration and process using multi-stage YAML templates and variable groups

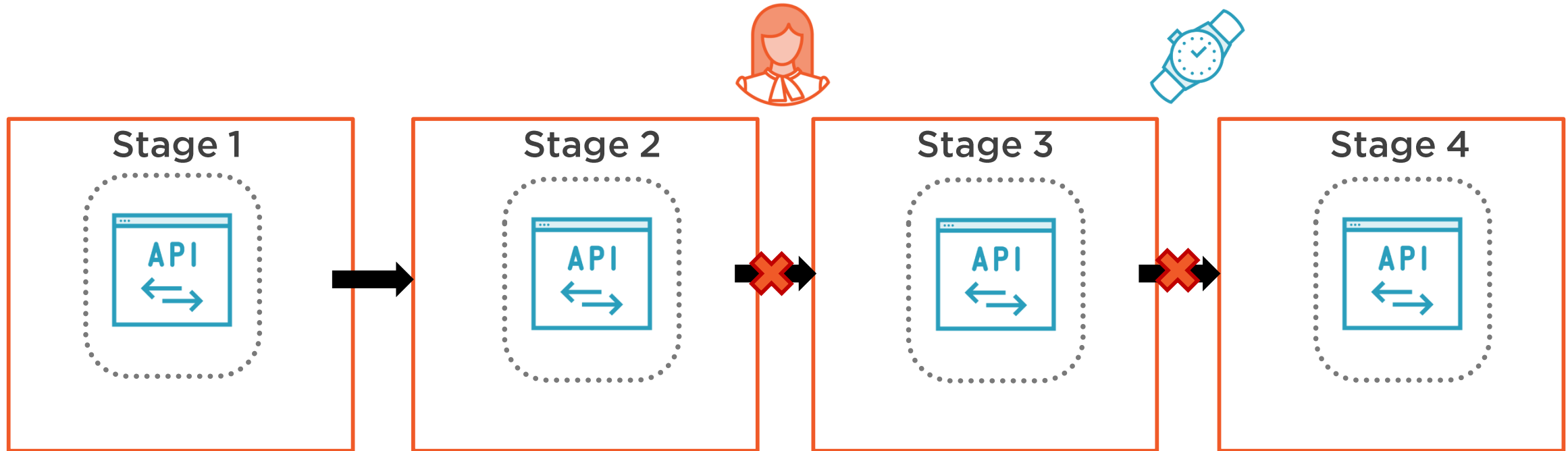
Summary



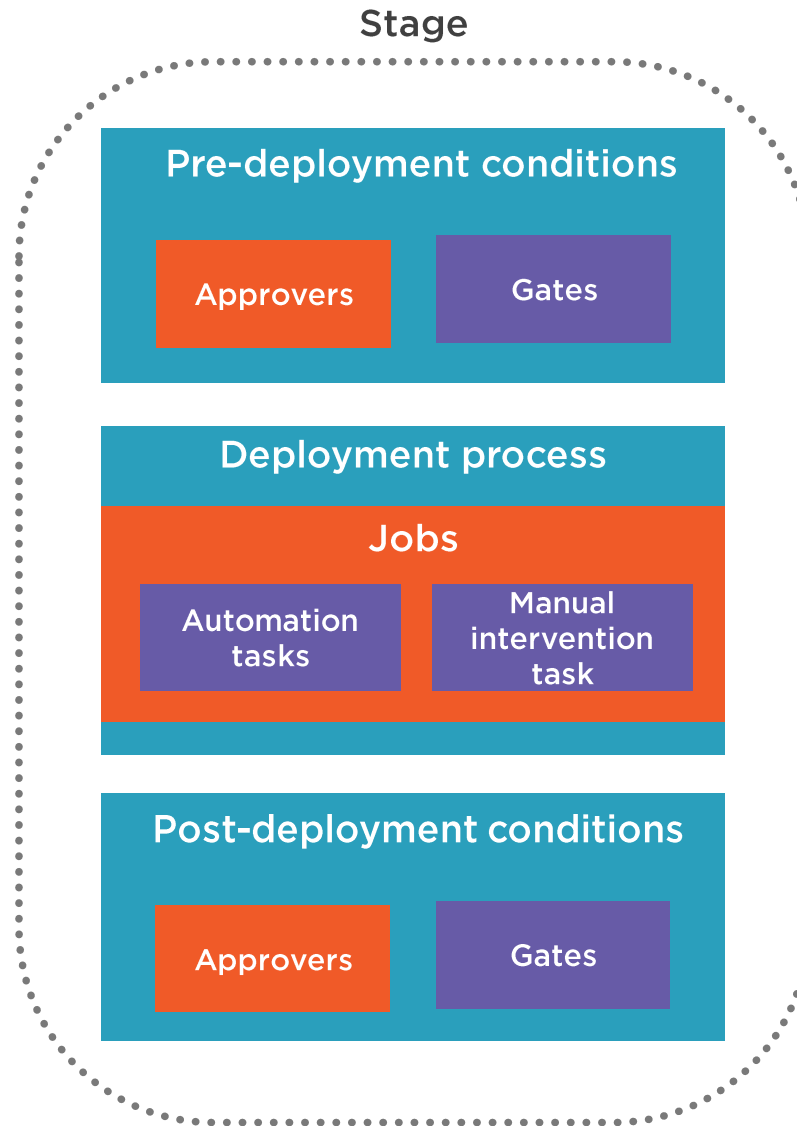
Organize Shared Release Process with Gates and Approvals



Gates and Approvals



Gates and Approvals



Approvals and gates give you additional control over the start and completion of the release pipeline



Scenarios for Gates and Approvals



You want to ensure there are no incidents from the monitoring or incident management system for the app after deployment



Some users must manually validate the change request and approve the deployment to a stage



During the deployment pipeline, you want to prompt the user to enter a value for a parameter used by the deployment tasks



Ensure the required status for work items, incidents, and issues is set



Notify non-Azure Pipelines users such as legal approval departments, auditors, or IT managers



Demo



Implement release gates and approvals

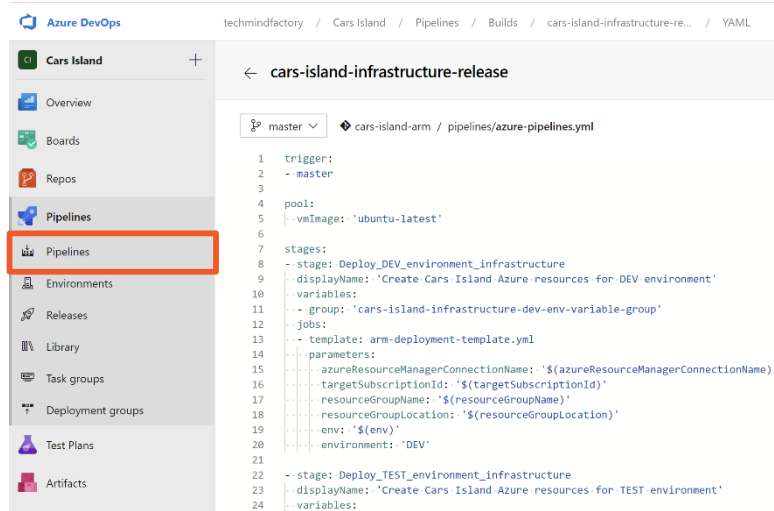
- Update current release with gates and approvals



Organize Releases Using Multi-stage YAML Templates



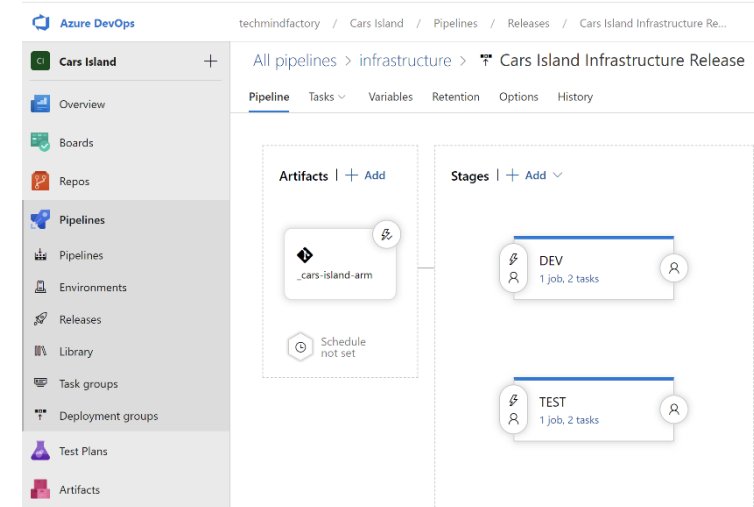
Azure DevOps Pipelines and Releases



The screenshot shows the Azure DevOps web interface for editing a pipeline. The left sidebar contains a navigation menu with items like Overview, Boards, Repos, Pipelines, Environments, Releases, Library, Task groups, Deployment groups, Test Plans, and Artifacts. The 'Pipelines' item is highlighted with an orange rectangle. The main area displays the 'cars-island-infrastructure-release' pipeline in YAML format. The YAML code defines a pipeline with a trigger on the 'master' branch, a pool using 'ubuntu-latest', and two stages: 'Deploy_DEV_environment_infrastructure' and 'Deploy_TEST_environment_infrastructure'. Each stage includes a job that uses an ARM deployment template.

```
1 trigger:
2   - master
3
4 pool:
5   - vmImage: 'ubuntu-latest'
6
7 stages:
8   - stage: Deploy_DEV_environment_infrastructure
9     displayName: 'Create Cars Island Azure resources for DEV environment'
10    variables:
11      - group: 'cars-island-infrastructure-dev-env-variable-group'
12    jobs:
13      - template: arm-deployment-template.yml
14        parameters:
15          azureResourceManagerConnectionName: '$(azureResourceManagerConnectionName)'
16          targetSubscriptionId: '$(targetSubscriptionId)'
17          resourceGroupName: '$(resourceGroupName)'
18          resourceGroupLocation: '$(resourceGroupLocation)'
19          env: '$(env)'
20          environment: 'DEV'
21
22   - stage: Deploy_TEST_environment_infrastructure
23     displayName: 'Create Cars Island Azure resources for TEST environment'
24     variables:
```

Pipelines (YAML files)

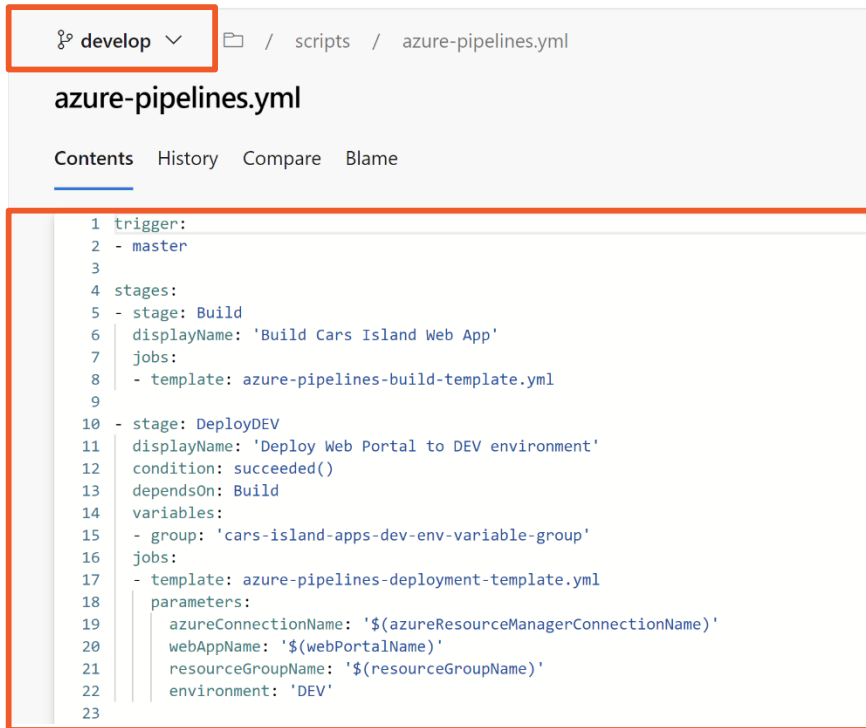


The screenshot shows the Azure DevOps Releases Designer interface. The left sidebar is the same as the previous screenshot, with 'Releases' highlighted. The main area shows the 'Cars Island Infrastructure Release' pipeline. It features a visual representation of the pipeline with an artifact named '_cars-island-arm' and two stages: 'DEV' and 'TEST'. Each stage is represented by a box showing its name and the number of jobs and tasks. The 'DEV' stage has 1 job and 2 tasks, while the 'TEST' stage has 1 job and 2 tasks. The interface also includes tabs for Pipeline, Tasks, Variables, Retention, Options, and History.

Releases (Designer)



Define Pipelines Using YAML Syntax



```
1 trigger:
2 - master
3
4 stages:
5 - stage: Build
6   displayName: 'Build Cars Island Web App'
7   jobs:
8   - template: azure-pipelines-build-template.yml
9
10 - stage: DeployDEV
11   displayName: 'Deploy Web Portal to DEV environment'
12   condition: succeeded()
13   dependsOn: Build
14   variables:
15   - group: 'cars-island-apps-dev-env-variable-group'
16   jobs:
17   - template: azure-pipelines-deployment-template.yml
18     parameters:
19       azureResourceManagerConnectionName: '$(azureResourceManagerConnectionName)'
20       webAppName: '$(webAppName)'
21       resourceGroupName: '$(resourceGroupName)'
22       environment: 'DEV'
23
```

Build and release pipelines can be defined in a YAML file called **azure-pipelines.yml**

YAML files can be stored in the source code repository



YAML Files



The pipeline is versioned together with the application's source code



Validation of the changes is done through code reviews in pull requests



Every branch you use can modify the YAML pipeline file and any change might cause a break in the release process



Demo



Implement release configuration and process using YAML files

- Use variable groups to keep shared configuration
- Setup environments
- Implement multi-stage YAML files
- Apply gates and approvals



Summary



How to implement gates and approvals for the release process

How to organize shared release process with multi-stage YAML files

How to design the release pipeline to ensure reliable order of dependency deployments



Thank you!



Daniel Krzyczkowski

MICROSOFT MVP & SOFTWARE DEVELOPER

@DKrzyczkowski www.techmindfactory.com

